

The 2006 TC-STAR Evaluation of the IBM Text-to-Speech Synthesis System

**Raul Fernandez, Raimo Bakis, Ellen M. Eide,
Wael Hamza, John F. Pitrelli, and Michael A. Picheny**

IBM T.J. Watson Research Center
Human Language Technologies – Text to Speech
Yorktown Heights, NY 10598
{fernandra, bakis, eide, hamzaw, pitrelli, picheny}@us.ibm.com

Abstract

In this paper we present the evaluation of the IBM TTS system completed within the 2006 TC-STAR evaluation. Speech corpora from three speakers and two languages (two Castilian Spanish speakers and one UK English speaker) are used to develop concatenative TTS systems, and subjective measures are gathered to assess the performance on three major tasks: evaluation of the prosody generation component of the TTS system, full-system evaluation using clean-text inputs, and full-system evaluation using the outputs of ASR and SLT systems. We report very encouraging results for what is the first evaluation using the datasets provided. In general, the Spanish systems outperform the English system, with the most optimistic evaluation figure finding the performance of the synthetic male Spanish voice to be very close to the performance attained by natural speech. The evaluation also supports the feasibility of using TTS to convey word-level understanding of recognized and translated input, with the best system, based on recognized English input translated to Spanish, achieving a 3.0% human-transcription word-error-rate on synthesized speech.

1. Introduction

In this paper we discuss the 2006 evaluation of the IBM Text-to-Speech (TTS) system carried out within the TC-STAR project, a research endeavor which aims to integrate Automatic Speech Recognition (ASR), Spoken Language Translation (SLT) and TTS technologies to produce Speech-to-Speech translation of unconstrained conversational speech in UK English, Castilian Spanish and Mandarin Chinese (TC-STAR, 2006). English and Spanish are the focus of the evaluations discussed in this paper; a companion paper submitted to this workshop addresses the evaluation of Mandarin using the IBM Mandarin TTS system (Jiang et al., 2006). The paper is organized as follows: In Section 2. we describe the tasks that the system is evaluated on, and the corpora that have been used for development. Section 3. describes the process of constructing the concatenative corpus for synthesis (3.1.) and the run-time system (3.2.). Finally, in Section 4., we present and discuss the results of the evaluation.

2. The TC-STAR Baseline Voice Corpora and Evaluation Subtasks

For each of the languages considered in this evaluation, a baseline voice corpus of approximately 10 hours of professionally-read, native speech was produced following the specifications detailed in Bonafonte et al. (2005) (these specifications address issues of phonetic coverage, coverage of representative in-domain material, etc.). Among the several objectives of this campaign was to evaluate the performance of full TTS systems under a variety of conditions, as well as the performance of TTS modular components, namely text processing, prosody generation, and acoustic synthesis. For English and Spanish, the IBM submissions investigated the performance of the prosody module and of the full systems. The data exchange format for the prosody component is an SSML file, reflecting the outputs of the text-processing module, and to which prosody specifications are added. The prosody description for an

utterance consists of (i) fundamental-frequency values for voiced sounds specified at 5-ms. intervals, (ii) duration values specified at the phone level, (iii) energy values specified at the phone level, (iv) primary and secondary stress specified at the syllable level, and (v) minor and major breaks following a word boundary. The energy and stress value specifications were not used when generating the stimuli for the evaluation of the prosody module. We next review the pertinent evaluation tasks:

- Task M2: Evaluation of the prosody module. This task consists of the following evaluation subtasks:
 - Task M2.1: Evaluation of prosody using segmental information: A natural utterance from the same speaker who recorded a dataset is resynthesized according to the prosody description output by the module, and the quality of the prosody judged on a 5-point scale.
 - Task M2.2: Judgment test using delexicalized utterances: To retain only the prosodic characteristics, an utterance produced as in M2.1 is delexicalized by rendering unvoiced sounds as silences and reproducing voiced sounds with only two harmonic sinusoidal functions (Bonafonte et al., 2005). Subjects are provided with a text transcript of the utterance and listen to the delexicalized stimulus before judging, on a 5-point scale, how well the prosody fits the text.
 - Task M2.3: Functional test using delexicalized utterances: For each utterance, produced as in M2.2, subjects choose which of 5 given sentences (texts) best corresponds to the prosody being heard.
- Task S1: Full system evaluation. Clean text inputs are used to synthesize acoustic waveforms. Subjects evaluate on a 5-point scale the following attributes of the synthesized speech: overall quality, listening effort,

pronunciation, comprehension, articulation, speaking rate, naturalness, pleasantness, and audio flow.

- **Task S2:** Full system evaluation using outputs from ASR and SLT modules: Speech that has been recognized and translated to a target language by the ASR and SLT modules respectively is synthesized. Subjects transcribe what they hear, and the subjective transcription is aligned with the input to the TTS system to compute errors.

3. System Description

Our speech synthesis system adopts a concatenative text-to-speech approach (Donovan and Eide, 1998; Syrdal et al., 2000; Coorman et al., 2000). The construction of the basic speech corpus is described in Section 3.1., and the basic run-time system TTS system is described in Section 3.2.

3.1. Constructing the Concatenative Voice Corpus

Preprocessing Acoustic Signals The recordings were produced on a platform with two synchronized channels to obtain (i) acoustic waveforms captured with a large membrane microphone and (ii) laryngograph signals. The speech recordings, originally sampled at 96 kHz and encoded at 24 bits per sample, were downsampled to 22.05 kHz and encoded at 16 bits per sample to provide consistency with the sampling rate used in our standard procedure to obtain automatic alignments. The speech was also high-pass filtered at 75 Hz and automatically annotated with instances of glottal closure for later processing. The laryngograph signals were not used in detecting glottal instances, although that is a possible variant to explore in the future.

Aligning Text to Speech A rule-based front end, described in Section 3.2., was used to generate a dictionary which typically contains one pronunciation for each of the words in the script. This basic dictionary was then augmented with pre-existing standard dictionaries to produce a final pronunciation dictionary used to provide phone sequences to align to the text. Combining the pronunciations produced by the front end with those of pre-existing dictionaries ensures that every word in the script will have at least one pronunciation, yet allows the flexibility of multiple pronunciations for most of the words to cover common alternate pronunciations. In the case of Castilian Spanish, the pre-existing dictionary used was the LC-STAR Spanish lexicon (LC-STAR, 2004) whereas for UK English, an internal IBM dictionary was used.

The acoustic signals were initially encoded as 12-dimensional mel-frequency cepstral coefficients plus log energy, as well as their first and second differences. The resulting 39-dimensional feature vector was extracted from 25-ms frames at a uniform 10-ms frame rate. Then, forced alignment using these acoustic features and a set of speaker-independent hidden Markov models (HMMs) generated an initial time-aligned phonetic transcription of the speech. These HMMs are mostly three-state left-to-right models, except for those used to model voiced plosives, which produce better alignments with just two states.

The acoustic features were then re-computed using 6-ms frames at a uniform 3-ms rate throughout regions of unvoiced speech, and pitch-synchronously throughout voiced regions with 25-ms frames centered on each glottal closure location computed earlier (Donovan, 1996). This re-coding provides better resolution through voiceless regions and helps when segmenting plosives due to the short timescales and rapid transitions involved. After two intermediate iterations of training, the models were used to arrive at a final alignment of the corpus. Each state-sized sub-phonetic waveform portion aligned with each state of an HMM in this final alignment was prepared to be used as synthesis token, the main building block used during synthesis.

Building Acoustic Models From these final alignments, acoustic trees (Breiman et al., 1984) were built to cluster the tokens of each HMM state and therefore reduce the eventual number of segment candidates over which the unit search is carried out. Each token associated with a given state was first transformed into a sequence of acoustic feature vectors using the pitch-synchronous procedure described earlier. Each tree was allowed to grow by computing maximum likelihood scores based on these acoustic features, and making splits at each node based on questions about the token's phonetic context (Bahl et al., 1993).

Precomputing Concatenation Costs To determine a concatenation cost for run-time that reflects spectral continuity between tokens, the IBM system makes use of a context-dependent distance measure between perceptually-modified cepstral vectors, as detailed previously (Donovan, 2001). The tokens were first segmented according to the voicing-dependent frame rate described above, and a 12-dimensional vector of perceptually-modified mel-frequency cepstral coefficients was extracted for each frame. For each state boundary transition, a context vector was assembled containing the identity of the current phone as well as either the boundary location for within-phone boundaries, or the identity of the preceding and following phones for cross-phone boundaries. These context vectors were paired with observation vectors consisting of the difference between the last perceptual cepstral vector belonging to the pre-boundary token and the initial cepstral vector of the post-boundary token. A decision tree was then grown using the phonetic context to cluster observation vectors, and the mean vector and diagonal covariance matrix associated with the tokens gathered at each leaf were recorded for further use in the search cost function at synthesis time.

Building Prosody Models The goal of prosody models for pitch, duration and energy is to predict target values for these variables that are used at run time to drive the search for suitable segments and, optionally, used to modify the output speech to match these targets. In our system we opt not to modify to the prosodic targets. Rather than imposing the predicted contours in the output, we derive our final pitch and duration contours from the actual prosody in the segments selected by the search.

In the most general case, the IBM TTS system allows the flexibility of several style-specific prosody models that can be switched on and off at run time depending on an attribute value associated with the style variable. We use the term

style here somewhat generically to refer to a quality of the speech data (linguistic, phonetic, expressive, etc.) that can be used to treat different portions of the synthesis differently. This style framework is flexible enough to allow several applications. It constitutes, for instance, the basis of one approach to generate expressive synthesis in the IBM TTS system (Pitrelli et al., 2006).

For the work reported in this paper, we have explored a novel approach to building prosody models by treating the phrase-finalness of a word in the text as a style variable, and building different pitch prediction models for each of the two styles, phrase-final and non-phrase-final. Although previously model the distance from the end of the phrase was used as a feature in the decision tree, so that theoretically the phrase-final distinction could have been captured, the relatively small number of observations occurring at the end of a phrase may have caused phrase-final pitch trends to be lost. To preserve those trends better, we divide our training observations into two categories, phrase-final and non-phrase-final. All syllables for all phrase-final words were used to train the phrase-final model; all other observations were used for the non-phrase-final model. Decision trees (Breiman et al., 1984) for predicting target f_0 values during synthesis for each of these categories were built based on a set of features extracted from the output of the front-end text analysis. Each tree predicts a vector of three f_0 values for the beginning, middle, and end of the sonorant region associated with a syllable, where the “middle” is defined as the center of the syllable’s most-sonorous phone. That is, we model f_0 within the syllable nucleus plus any adjacent nasals, liquids and glides within that syllable.

The basic set of f_0 -predictor features includes text-based information such as lexical stress of the current syllable, and part-of-speech of the current word. A set of 14 features per syllable was extracted over a context window of five syllables — the current syllable plus the two preceding and two following syllables — and the feature vectors stacked to form the full set of features used to grow the tree. A multivariate Gaussian was used to model the distribution of log f_0 observations at each node, where f_0 is in Hz.

Once the phrase-final and non-phrase-final trees were built, the data used to build each tree were traversed down that tree. The mean of the feature vectors mapped to each leaf forms the prediction vector for that leaf.

For modeling duration, a single decision tree was built to predict the log duration of each phone to be synthesized. Although we could have followed an approach analogous to the pitch modeling described above, and built different duration trees based on the phrase-finalness of the data, time constraints did not allow us to experiment with this. This is, however, a good idea to pursue further, as some known phenomena (*e.g.*, pre-boundary segmental lengthening), could be better modeled with this approach. We plan to verify these hypotheses by building phrase-final specific duration trees and running formal listening tests.

The set of features used for duration modeling is similar to the set used to predict f_0 . A tree was built from these feature vectors assuming a Gaussian distribution to model the log-duration observations at each node of the tree.

The trees for predicting f_0 and duration targets could be

complemented by energy-prediction decision trees. We have empirically found, however, that the contribution of the energy models to the quality is negligible, and so this component of the system is often disabled at run time, including in the evaluation tasks described in Section 2.

Allowing Alternate Pronunciations Perceived quality often improves when the synthetic speech includes longer spans of pre-recorded material with fewer splices, thereby better preserving the professional speaker’s original naturalness. Populating the script with commonly-used words and phrases for the particular domain of the application (such as has been the case for the TC-STAR corpora) could enable the run-time search to find whole, contiguous instances of such material, and avoid having to concatenate them from smaller segments. One obstacle, however, occurs when there is a mismatch between the speaker’s pronunciation of a word and the pronunciation proposed by the front end. An algorithm, previously described in more detail (Hamza et al., 2004), reconciles these mismatches by augmenting the search space using a dictionary of alternate pronunciations which is assembled during the corpus-construction process, and includes spellings, a list of tokens comprising a word (as pronounced by the speaker), and the context in which these tokens occur.

3.2. Run-Time Full System Output

Front End The goal of the “front end” of our system is to process the input text in order to determine the words and phones to be spoken, as well as other information, such as part-of-speech tagging and pause placement, which will influence the prosody to be chosen in later stages of synthesis. The front end is rule-based, and divided into four tasks. The first, “text normalization,” determines the sequence of actual words to be said, by expanding abbreviations and processing numbers, acronyms, and special items such as URLs and e-mail addresses. Second, a parser is applied to this word sequence, tagging words with parts of speech, and determining phrase structure. Third, a “morpheme analyzer” identifies prefixes, suffixes, and the root, or roots in the case of compounding. Finally, the “phonetizer” proposes a sequence of phone labels, syllables and stress levels to represent the input text. The output of the front end is further augmented at this stage with any style labels (at the word level), specifying any attributes that will be used by the prosody modules to switch between different models. For the experiments reported here, each word found to precede a phrase boundary was given a “phrase-final” label. (Unlabeled words are accorded an externally specifiable default status, in this case “non-phrase-final.”)

Prosody Models During synthesis, the front end output is parsed; text-based features for each syllable and phone in the sentence to be synthesized are assembled, and the corresponding style-specific prosody model is invoked. The features are used to traverse the f_0 and duration trees, respectively, until a final leaf is reached.

The mean values of the observed f_0 contours gathered at each leaf are used to construct the target f_0 contour. The estimated f_0 contour of the sonorant region within the syllable is then linearly interpolated; the resulting contour is

used to evaluate the f_0 -target component of the cost function for each candidate token at search time.

Similarly, the mean of the log duration of all tokens in the appropriate leaf in the duration tree is used as the target for the phone to be synthesized.

Generating the Sequence of Tokens for Concatenation

The sequence of N phones generated by the front end, $[p_1 \cdots p_N]$ is used to generate a sequence of acoustic leaves using the acoustic decision trees grown earlier. First, each phone, p_i , is split into a number of states, $[s_1 \cdots s_{N_i}]$, where N_i is the number of states in the HMM that corresponds to phone p_i . Then, the tree for each state is traversed, generating a sequence of leaves $[l_1 \cdots l_M]$, where M is the number of leaves/states in the whole sentence. As described in Section 3.1., a phonetic-context feature vector is used to traverse the tree. The resulting leaves are considered to be categories of tokens in the concatenative speech synthesis framework.

Forming the Target Sequence For each leaf l_j in the leaf sequence $[l_1 \cdots l_M]$, a target specification A_j is created and is used afterward in the token search. Each target specification A_j contains information about the predicted duration and f_0 . The target durations for leaves corresponding to phone p_i are calculated as follows. The medians of duration for each leaf corresponding to phone p_i are recalled. The ratio between those leaf duration medians is used to distribute among the leaves the duration of p_i predicted by the phone prediction module described earlier. Start f_0 and end f_0 of each leaf l_j are calculated by sampling the predicted f_0 contour on points corresponding to the start and the end of each leaf.

Forming the Search Trellis For each leaf target A_j in the target sequence, all tokens that belong to the same leaf l_j are recalled to be candidates for selection. Each candidate O_j^k , which refers to the j -th leaf's k -th token, for target A_j contains information about its duration, its start and end f_0 , and its boundary perceptual cepstral vectors. A search trellis is formed out of all token candidates belonging to the sequence of leaf targets.

In addition, each word in the word sequence is looked up in the alternates dictionary and the list of token candidates corresponding to that word is retrieved (Hamza et al., 2004) in order to handle alternate pronunciations as described in Section 3.1. For each leaf in the leaf sequence, the list of candidate tokens is augmented with the list of tokens coming from the alternates dictionary which meet context constraints requiring that, for an alternate token to be considered for within the first (last) phone of a word, the phone preceding (following) that token's phone in the corpus must match the last (first) phone of the preceding (following) word in the synthesized sentence. Furthermore, we only consider alternates that share the same number of subphonetic units with the original word, in order to be able to use the same search trellis (just with the augmented forms). Alternates with a different number of phones was considered, but eliminated due to the computational complexity of the graph search that would need to be implemented. When alternates are considered, the prosody targets are still predicted based on the contextual features of the original word.

Search A Viterbi beam search is used to search the candidate trellis, similarly to other concatenative TTS systems (Hunt and Black, 1996). The search aims to select the token sequence with the least cost from among the candidate tokens. Our cost function is defined as follows:

$$C \left(\left\{ A_j; O_j^{k_j} \right\}_{j=1}^M \right) = \sum_{j=1}^M C_{ta} \left(A_j, O_j^{k_j} \right) + \sum_{j=2}^M C_{co} \left(O_{j-1}^{k_{j-1}}, O_j^{k_j} \right) \quad (1)$$

where $C_{ta} \left(A_j, O_j^{k_j} \right)$ is the target cost between target A_j and candidate token $O_j^{k_j}$, $C_{co} \left(O_{j-1}^{k_{j-1}}, O_j^{k_j} \right)$ is the concatenation cost between candidate tokens $O_{j-1}^{k_{j-1}}$ and $O_j^{k_j}$, and M is the number of segments in the sequence.

The target cost comprises a set of cost components namely,

- the f_0 cost, measuring how far the token's f_0 contour is from that of the target (*i.e.*, the absolute value of the difference between the f_0 of the target and that of the token, divided by the smaller of the two). This is evaluated both at the start and the end of each token.
- the duration cost, which measures how much the duration of the token deviates from the target,

The concatenation cost comprises two components, namely,

- the context-dependent Mahalanobis distance between the last 12-dimensional vector of $O_{j-1}^{k_{j-1}}$ and the first vector of $O_j^{k_j}$ (Donovan, 2001) and
- the f_0 transition cost between the end f_0 value of $O_{j-1}^{k_{j-1}}$ and the start f_0 value of $O_j^{k_j}$.

The target cost components and the concatenation cost components are added using weights tuned empirically.

Generating the Output Speech

Pitch Processing Concatenating speech tokens without processing might result in mismatches, unnatural jumps in f_0 , and unwanted "pitch warble." To remove such effects, a new target contour is generated by convolving the concatenated $f_0(t)$ with a smoothing kernel $h(\tau)$, and the signal is then processed to match this new contour. The kernel must be even, in order to introduce no phase shifts and also to preserve the normal pitch variations while smoothing out discontinuities. We chose a two-sided exponential $h(\tau) = \frac{1}{2\tau_0} e^{-|\tau/\tau_0|}$, where τ_0 is a time constant, typically in the range (0.02 – 0.07) sec. This produces generally pleasing f_0 contours, and is computationally efficient. Within each segment, f_0 is either linearly interpolated to produce a highly smooth contour, or the microprosody is preserved as described later in this section, to obtain a somewhat more natural contour, preserving some short-term characteristics of the speaker's original pitch. While a smooth contour is generally preferable to unintended rapid fluctuations, smoothing has undesired side effects. Sometimes, f_0 fluctuations are necessary to convey

Language	System	M2.1 (1-5)	M2.2(1-5)	M2.3 (0-1)
EN	IBM (F)	2.457	2.087	0.376
	NAT (F)	4.111	3.972	0.648
ES	IBM (M)	2.767	3.167	0.750
	IBM (F)	-	3.194	0.527
	NAT (M)	4.189	3.583	0.75

Table 1: Evaluation results of the prosody module evaluation task for UK English (EN) and Castilian Spanish (ES).

expressions such as excitement. Smoothing also reduces other expressive voice qualities, such as creakiness. To preserve such effects whenever possible, we bypass signal processing completely for sufficiently long sequences of tokens which had been contiguous in the original corpus, except near the edges of the sequence, where we impose a gradual transition to the f_0 of the adjacent segments. This “contiguous bypassing” approach allows maximally-faithful reproduction of expression present in the corpus.

In the results reported in this paper, we use a compromise between full f_0 smoothing and complete bypassing: smoothing with microprosody preservation. In this case, the f_0 targets at segment boundaries are calculated as they would be for full f_0 smoothing, but within each segment the original f_0 is multiplied by a linearly time-varying factor such that the result matches the smoothed targets at both ends of the segment. While the effect of microprosody preservation is small when the segments are short, it produces noticeably more natural contours in long segments than does linear interpolation.

Note that after the unit search, which made use of our original prosody targets in its cost function, those targets are discarded and the observed durations and the smoothed version of the observed pitches with micro-prosody preservation described in this section form the prosody contour of our synthesized waveform. It is this set of pitches and durations which we submitted as the output of our prosody model for the prosody evaluation.

Amplitude Compression Our system includes optional automatic gain control (AGC) and limiter functions applied as the last step after synthesis is complete (Zölzer, 1997). The gain control function is the logical equivalent of two cascaded gain controls: an AGC for dynamic-range compression, and a limiter for preventing overload. The function looks ahead three time samples to allow it to reduce gain in anticipation of large signal peaks, thus minimizing the need for sudden gain changes that might introduce clicks or distortion. We generally use only a small amount of dynamic-range compression, increasing the maximum gain by no more than 6 dB above what would be possible without peak limiting.

Downsampling The basic concatenative corpus was built using speech sampled at 22.05 kHz. In order to conform to the specifications of this evaluation, the final synthetic waveform produced by the procedure described in this section, is downsampled to 16 kHz, and encoded at 16 bits per sample, using MS-Win wave format.

4. Results and Discussion

Tables 1-3 summarize the evaluation results for the tasks outlined in Section 2. The output of the prosody module (M2 task) consists of the natural prosody of the segments selected by the search, followed by pitch smoothing. It is, in other words, (a smoothed version of) the prosody of segments in the database which best fit the overall search criterion (which includes, as described earlier, a spectral continuity criterion). While this makes most sense in a fully integrated system in which the same segments from which the prosody is taken are the ones used to generate the waveform, this approach of transplanting the prosody of the segments chosen by the search to resynthesize an arbitrary utterance still performs well in isolation. The system tasks (S1 and S2) are evaluated on the output of the full system described in this paper, with automatic gain control, contiguous bypass and pitch smoothing enabled. For the S1 task, two sets of results are presented, as the systems were evaluated a second time to include the later submission of an additional system.¹

As we can see from these tables, the Spanish voices outperform the UK English voice for all the tasks being evaluated. For the Spanish male voice, in particular, we obtained figures that are close to those reported for the natural voice (see, for instance, task M2.3 in Table 1, and the various figures reported in Table 2). The overall lesser performance obtained on the English voice may be due to characteristics of the corpus which we have now only begun to explore. We have observed, for instance, a degree of variability in the delivery style of the English speaker that may require a dataset larger than the current corpus to be handled appropriately. A second factor potentially impacting the performance may be the amount of in-domain material contained in the English corpus. Our system consistently produced higher percentage of unit splices when synthesizing English (on the order of 20%, versus 13% for Spanish), suggesting that the contiguous bypass feature of the search might be exploiting a richer inventory of in-domain material in Spanish, and extracting longer contiguous spans of unmodified units from the dataset. We need to point out that the female Spanish voice, based on the same script, does not achieve the same quality as the male speaker; so clearly the nature of the script interacts with the delivery when contributing to the quality. Here, we have observed a more consistent and affectless style in the male speaker, suggesting, once again, the advantage of a more restrained delivery style for

¹No figure was reported for the resynthesis task (M2.1) for the female Spanish speaker, as there was no natural utterance available to carry out the evaluation.

Lang.	Eval	System	A	B	C	D	E	F	G	H	I	J
EN	S1.I	IBM (F)	3.417	3.630	3.546	3.944	3.787	4.426	2.417	2.713	3.176	2.704
		NAT (F)	4.581	4.591	4.772	4.844	4.613	4.670	4.481	4.409	4.313	4.330
	S1.II	IBM (F)	3.130	3.676	3.639	3.787	3.500	4.111	3.065	2.898	3.148	2.741
		NAT (F)	4.792	4.906	5.000	4.978	4.950	4.789	4.617	4.481	4.481	4.661
ES	S1.I	IBM (M)	4.333	4.611	4.556	4.694	4.306	4.472	3.861	3.889	4.000	3.444
		IBM (F)	3.556	4.333	4.111	4.694	4.139	4.528	3.333	3.500	3.722	4.333
		NAT (M)	4.611	4.889	4.889	4.944	4.667	4.917	4.583	4.361	4.278	4.333
	S1.II	IBM (M)	4.333	4.361	4.194	4.556	4.389	4.639	3.667	3.778	3.972	3.806
		IBM (F)	3.917	3.917	3.694	4.500	4.250	4.028	3.000	3.278	3.472	3.139
		NAT (M)	4.656	4.844	4.875	4.875	4.875	4.719	4.531	4.438	4.438	4.438

Table 2: Evaluation results for the S1 task for UK English and Castilian Spanish for two rounds of evaluation. Indicated in parenthesis is the gender of the speaker for the IBM system and natural (NAT) speech. The following attributes of speech are being evaluated on a scale from 1-5: A=Overall Quality; B=Listening Effort; C=Pronunciation; D=Comprehension; E=Articulation; F=Speaking Rate; G=Naturalness; H=Ease of Listening; I=Pleasantness; J=Audio Flow.

Language	Spkr.	# Sent	# Words	% Corr.	% Sub	% Del	% Ins.	WER	SER
EN	F	55	662	94.0	3.9	2.1	1.2	7.3	47.3
ES	M	76	969	97.5	1.9	0.6	0.5	3.0	25.0
	F	74	933	96.1	2.9	1.0	0.5	4.4	33.8

Table 3: Evaluation results for the S2 task for UK English and Castilian Spanish.

a corpus of limited size.

The word error rate (WER) and sentence error rate (SER) figures shown in Table 3 suggest the feasibility of using TTS to generate speech that preserves word-level understandability in the presence of errors introduced by ASR and SLT systems. An important future research direction would be to improve this performance by, for instance, using more sophisticated prosodic models that help with the understanding of potentially unpredictable content. Additionally, we face the task of evaluating content understanding, a goal which remains at the core of a speech-to-speech translation system.

5. References

- L.R. Bahl, P.V. deSouza, P.S. Gopalakrishnan, and M.A. Picheny. 1993. Context dependent vector quantization for continuous speech recognition. In *Proc. ICASSP*, Minneapolis, MN, U.S.A.
- A. Bonafonte, H. Höge, H.S. Tropf, A. Moreno, H. van der Heuvel, D. Sündermann, U. Ziegenhain, J. Pérez, and I. Kiss. 2005. TTS baseline and deliverables. Technical report, Technology and Corpora for Speech to Speech Translation (TC-STAR) <http://www.tc-star.org>.
- L. Breiman, J.H. Friedman, R.A. Olshen, and J. Stone. 1984. *Classification and Regression Trees*. Wadsworth.
- G. Coorman, J. Fackrell, P. Rutten, and B. Van Coile. 2000. Segment selection in the L&H RealSpeak Laboratory TTS system. In *Proc. ICSLP*, Beijing, China.
- R. E. Donovan and E. Eide. 1998. The IBM trainable speech synthesis system. In *Proc. ICSLP*, Sydney, Australia.
- R.E. Donovan. 1996. *Trainable Speech Synthesis*. Ph.D. thesis, Cambridge University Engineering Department.
- R.E. Donovan. 2001. A new distance measure for costing spectral discontinuities in concatenative speech synthesizers. In *Proc. 4th ISCA Tutorial and Research Workshop on Speech Synthesis*, Atholl Palace Hotel, Scotland, U.K.
- W. Hamza, R. Bakis, E. Eide, M. A. Picheny, and J. F. Pitrelli. 2004. The IBM expressive speech synthesis system. In *Proc. ICSLP*, Korea.
- A. Hunt and A. Black. 1996. Unit selection in a concatenative speech synthesis system using a large speech database. In *Proc. ICASSP*, Atlanta, GA, U.S.A.
- D. Jiang, Q. Shi, F. Meng, Shuang Z., X. Ma, Y. Liu, and Qin. Y. 2006. Overview of the IBM mandarin Text-to-Speech system. In *Workshop on Speech-to-Speech Translation*, Barcelona, Spain, June.
- LC-STAR. 2004. Lexica and corpora for Speech-to-Speech translation components (<http://www.lc-star.com>).
- J. F. Pitrelli, R. Bakis, E. M. Eide, R. Fernandez, W. Hamza, and M. A. Picheny. 2006. The IBM expressive Text-to-Speech synthesis system for American English. *IEEE Trans. Audio, Speech and Language Processing*, July.
- A. K. Syrdal, C. W. Wightman, A. Conkie, Y. Stylianou, M. Beutnagel, J. Schroeter, V. Strom, K. Lee, and M.J. Makashay. 2000. Corpus-based techniques in the AT&T NextGen Synthesis System. In *Proc. ICSLP*, Beijing, China.
- TC-STAR. 2006. Technology and corpora for Speech-to-Speech translation (<http://www.tc-star.org>).
- U. Zölzer. 1997. *Digital Audio Signal Processing*. J. Wiley & Sons, Chichester.